

IN THE CLAIMS

- 1 1. [currently amended] A process comprising:
2 validating a ~~formal language specification~~ Formal Language Specification written
3 in a formal language which has predetermined rules of syntax and semantics, said ~~formal~~
4 ~~language specification~~Formal Language Specification defining a computer program to be
5 automatically written;
6 automatically translating each element of a ~~formal language specification~~ Formal
7 Language Specification defining an object model, a functional model, a dynamic model
8 and a presentation model, which taken together define the requirements of the program
9 to be automatically written, into a full and complete computer program which needs no
10 additional third party source code or source code from existing components or code
11 libraries to be compiled with it to make said computer program complete and which
12 implements the requirements of said ~~formal language specification~~ Formal Language
13 Specification, said ~~formal language specification~~ Formal Language Specification defining
14 at least an identification function for every class, and at least a valuation for every
15 variable attribute, said translating step comprising the following steps:
16 using a computer, automatically write computer code that will request user
17 name and password, receive any responses and authenticate the user;
18 using a computer, automatically write computer code that has the
19 capability to control a computer to provide a mechanism to will determine a this
20 user's privilege level from log in information supplied by said user which identifies
21 said user and query said ~~formal language specification~~ Formal Language
22 Specification and determine all object attributes said user has privilege to see and
23 all services said user has privileges to can invoke;
24 using a computer, automatically write computer code which has the
25 capability to query queries said ~~formal language specification~~ Formal Language
26 Specification for all services of all classes that any authorized user has privileges
27 to may invoke and identify identifies an object server which will implement each
28 said service;
29 using a computer, automatically write code that has the capability to will
30 retrieve service arguments for all services;
31 using a computer, automatically write code that is capable of controlling
32 controls a computer to provide a display means by which an entity has a

33 mechanism to and entity can invoke a service, and which has a mechanism to
34 receive receives input to invoke a particular service and respond responds by
35 sending a message to the appropriate object server to invoke the service, said
36 message including the necessary arguments required for the service to execute;
37 using a computer, automatically write code that has the capability to
38 control a computer to implement implements an object server for every service,
39 each of which first checks to verify that state transitions are valid and make
40 sense for the current state of objects of which the object server will be altering
41 the state;
42 using a computer, automatically write code that has the capability to
43 control a computer to implement for every object server that verifies preconditions
44 are satisfied before making state transitions of any objects the states of which
45 are acted upon by the object server;
46 using a computer, automatically write code that has the capability to
47 control a computer to make all valuation calculations required by said formal
48 language specification Formal Language Specification of each object server;
49 using a computer, automatically write code that has the capability to
50 control a computer to verify that integrity constraints specified in said formal
51 language specification Formal Language Specification on the values of attributes
52 of objects have been satisfied after execution of a service and respond by
53 reversing any changes in state which caused said integrity constraint to be
54 violated take-action if said integrity constraints are not satisfied; and
55 using a computer, automatically write code that has the capability to
56 control a computer to implement for every object server such that said object
57 server tests to test trigger relationships specified in said formal language
58 specification Formal Language Specification after execution of a service and
59 invoke a predetermined service associated with a trigger event carry out
60 appropriate action if said a trigger event has occurred.

1 2. [currently amended] An apparatus for automatically translating a formal
2 language specification Formal Language Specification defining an object model, a
3 functional model, a dynamic model and a presentation model, which taken together define
4 the requirements of a computer program to be automatically written, said formal language

5 specificationFormal Language Specification being written in a formal language which has
6 predefined rules of grammar, said translating acting to convert said formal language
7 specificationFormal Language Specification into a computer program that implements the
8 requirements of said formal language specificationFormal Language Specification, said
9 formal language specificationFormal Language Specification defining at least an
10 identification function for every class and at least a valuation for every variable attribute,
11 said apparatus comprising:

12 a computer programmed with an operating system and one or more other
13 programs to cooperate with said operating system to control said computer to perform
14 the following functions:

15 A) using said predetermined rules of grammar to validate said formal
16 language specificationFormal Language Specification to ensure that said formal
17 language specificationFormal Language Specification is complete and correct;

18 B) automatically write computer code that will request user name and
19 password, receive any responses and authenticate the user;

20 C) automatically write computer code that will determine a user's privilege
21 level for a user identified by user name and password entered in response to the
22 query caused by the code written in step B, and query said formal specification
23 and determine all object attributes said user has privilege to see and query and all
24 services said user has privileges to can invoke;

25 D) automatically write computer code which queries said formal
26 specification for all services of all classes that any authorized user has privileges
27 to may invoke and identifies an object server which will implement said service;

28 E) automatically write code that will retrieve service arguments for all
29 services;

30 F) automatically write code that displays one or more user interface tools
31 which provide a mechanism can be used to invoke a service, and which provides
32 a mechanism to receive receives input to invoke a particular service and which
33 responds by sending a message to the appropriate object server to invoke said
34 service, said message including the necessary arguments required for said
35 service to execute;

36 G) automatically write code that implements an object server for every
37 service, each of which first checks to verify that state transitions are valid and

38 make sense for the current state of objects the object service will be altering the
39 state of;

40 H) automatically write code for every object server that verifies
41 preconditions are satisfied before making state transitions of any objects the
42 states of which are acted upon by said object server;

43 I) automatically write code to make all valuation calculations required by
44 said specification of each object server;

45 J) automatically write code to verify that integrity constraints specified in
46 said Formal Language Specification ~~formal specification~~ on the values of
47 attributes of objects have been satisfied after execution of a service and
48 reversing any changes in state which caused said integrity constraints to be not
49 satisfied take action if said integrity constraints are not satisfied; and

50 K) automatically write code for every object server to test trigger
51 relationships specified in said Formal Language Specification ~~formal specification~~
52 after execution of a service and invoke a predetermined service associated with
53 a trigger event carry out appropriate action if a trigger event has occurred.

1 3. [currently amended] A physical computer-readable storage media ~~medium~~
2 containing instructions for controlling a computer to automatically translate a ~~formal~~
3 language specification Formal Language Specification defining an object model, a
4 functional model, a dynamic model and a presentation model, which taken together define
5 the requirements of a computer program to be automatically written, said ~~formal~~ language
6 specification Formal Language Specification defining at least an identification function for
7 every class and at least a valuation for every variable attribute said ~~formal~~ language
8 specification Formal Language Specification written in a formal language having
9 predefined rules of grammar, by:

10 validating a ~~formal language specification~~ Formal Language Specification written in
11 a formal language which has predetermined rules of syntax and semantics, said
12 validation accomplished using said predetermined rules of syntax and semantics to
13 ensure said ~~formal language specification~~ Formal Language Specification is complete and
14 correct;

15 automatically writing computer code that will request user name and password,
16 receive any responses and authenticate the user;

17 automatically writing computer code that will determine a user's privilege level and
18 query said Formal Language Specification ~~formal specification~~ and determine all object
19 attributes said user has privilege to see and all services said user has privileges to can
20 invoke;

21 automatically writing computer code which queries said Formal Language
22 Specification ~~formal specification~~ for all services of all classes that any authorized user
23 has privileges to may invoke and identifies an object server which will implement said
24 service;

25 automatically writing computer code that will retrieve service arguments for all
26 services;

27 automatically write code that displays menus options, icons or creates any other
28 means or mechanism through ~~by~~ which a user or another process has the capability to
29 can invoke a service, and which provides a mechanism through ~~which receives~~ input to
30 invoke a particular service is provided and mechanisms to provide values for arguments
31 is provided and a mechanism is provided to construct and send responds by sending a
32 message to the appropriate object server to invoke the service, said message including
33 the necessary arguments required for the service to execute;

34 automatically writing code that implements an object server for every service,
35 each of which first checks to verify that state transitions are valid ~~and make sense~~ for
36 the current state of objects the object service will be altering the state of;

37 automatically write code for every object server that verifies preconditions are
38 satisfied before making state transitions of any objects the states of which are acted
39 upon by the object server;

40 automatically write code to make all valuation calculations required by said Formal
41 Language Specification ~~formal specification~~ of each object server;

42 automatically write code to verify that integrity constraints specified in said Formal
43 Language Specification ~~formal specification~~ on the values of attributes of objects have
44 been satisfied after execution of a service and reverse any state changes which have
45 been made which cause said integrity constraints to be not satisfied take action if said
46 integrity constraints are not satisfied; and

47 automatically write code for every object server to test trigger relationships
48 specified in said Formal Language Specification ~~formal specification~~ after execution of a
49 service and invoke a predetermined service associated with a trigger event carry out

50 appropriate action if a trigger event has occurred.

1 4. [currently amended] An apparatus for automatically translating a Formal
2 Language Specification written in any formal language defining a full and complete
3 Conceptual Model of a desired computer program to be automatically generated into a full
4 and complete source code which implements said desired computer program,
5 comprising:

6 a computer programmed with an operating system and one or more other
7 programs to cooperate with said operating system to control said computer to perform
8 the following functions:

9 reading all said primitives in said Formal Language Specification in any
10 order;

11 in any order, using a computer and said Formal Language Specification to
12 automatically generate computer code which has the capability to control a
13 computer to carry out generating one or more methods ~~comprised of computer~~
14 code which can control a computer to perform the following functions in an order
15 determined by an execution model:

16 determining if said Formal Language Specification requires user
17 authentication, and, if so, automatically writing computer code that will
18 request user name and password, receive any responses and
19 authenticate the user;

20 determining if said Formal Language Specification requires
21 determining a user privilege level, and, if so, automatically writing
22 computer code that will determine a user's privilege level and query said
23 Formal Language Specification and determine all object attributes said
24 user has privilege to see and determine all services this user has
25 privileges to ~~can~~ invoke;

26 determining if said Formal Language Specification defines
27 services, and, if so, automatically writing computer code which queries
28 said Formal Language Specification to determine all services that the
29 authenticated user has privileges to ~~may~~ invoke and which are defined in
30 said Formal Language Specification for all classes of objects said
31 authenticated user will be able to view and automatically writing an object

32 server for each said service which will implement said service upon
33 receipt of a service invocation message, each of said object servers
34 containing code which will perform the following functions in the following
35 order upon receipt of a service invocation message:

36 verify that one or more proposed state transitions are valid
37 ~~can be validly made~~ for the current state of any object(s) of which
38 said object server will be altering the state before actually altering
39 the state of said object(s);

40 verify that any preconditions of a service implemented by
41 an object server are satisfied before said object server will act
42 upon said one or more objects to make state transitions thereof in
43 carrying out said service,

44 ignoring said service invocation message if either said one
45 or more state transitions cannot be validly made for the current
46 state of any objects upon which said object server will be acting
47 or any said precondition is not satisfied;

48 if all said proposed transitions are valid ~~can be validly made~~
49 on said one or more objects upon which said object server will act
50 and if all said preconditions are satisfied, make all valuation
51 calculations of said object server required by said Formal
52 Language Specification;

53 verify that service execution by said object server did not
54 result in violation of one or more integrity constraints specified in
55 said Formal Language Specification on the values of attributes of
56 objects affected by execution of said service implemented by said
57 object server, and take corrective action if one or more of said
58 integrity constraints are not satisfied; and

59 after a valid change of state of an object acted upon by
60 said object server occurs, test trigger relationships or condition-
61 action rules specified in said Formal Language Specification and, if
62 any trigger event is satisfied, triggering a service specified in said
63 condition-action rule or trigger relationship.

1 5. [Cancelled]

1 6. [Cancelled]

1 7. [Cancelled]

1 8. [Cancelled]

2 9. [Cancelled]

1 10. [Currently amended] A process for automatically translating a Formal
2 Language Specification written in any formal language defining a full and complete
3 Conceptual Model of a desired computer program to be automatically generated into a full
4 and complete source code which implements said desired computer program,
5 comprising:

6 reading all said primitives in said Formal Language Specification in any
7 order;

8 in any order, using a computer and said Formal Language Specification to
9 automatically generate computer code that has the capability to control a
10 computer to implement generating one or more methods which perform the
11 following functions in an order determined by an execution model:

12 determining if said Formal Language Specification requires user
13 authentication, and, if so, automatically writing computer code that will
14 request user name and password, receive any responses and
15 authenticate the user;

16 determining if said Formal Language Specification requires
17 determining a user privilege level, and, if so, automatically writing
18 computer code that will determine a user's privilege level and query said
19 Formal Language Specification and determine all object attributes said
20 user has privilege to see and determine all services this user has
21 privileges to can invoke;

22 determining if said Formal Language Specification defines
23 services, and, if so, automatically writing computer code which queries
24 said Formal Language Specification to determine all services that the
25 authenticated user has privileges to may invoke and which are defined in
26 said Formal Language Specification for all classes of objects said
27 authenticated user will be able to view and automatically writing an object
28 server for each said service which will implement said service upon
29 receipt of a service invocation message, each of said object server

30 containing code which will perform the following functions in the following
31 order upon receipt of a service invocation message:

32 verify that one or more proposed state transitions are valid
33 ~~can be validly made~~ for the current state of any object(s) of which
34 said object server will be altering the state before actually altering
35 the state of said object(s);

36 verify that any preconditions of a service implemented by
37 an object server are satisfied before said object server will act
38 upon said one or more objects to make state transitions thereof in
39 carrying out said service,

40 ignoring said service invocation message if either said one
41 or more state transitions cannot be validly made for the current
42 state of any objects upon which said object server will be acting
43 or any said precondition is not satisfied;

44 if all said proposed transitions are valid ~~can be validly made~~
45 on said one or more objects upon which said object server will act
46 and if all said preconditions are satisfied, make all valuation
47 calculations of said object server required by said Formal
48 Language Specification;

49 verify that service execution by said object server did not
50 result in violation of one or more integrity constraints specified in
51 said Formal Language Specification on the values of attributes of
52 objects affected by execution of said service implemented by said
53 object server, and take corrective action if one or more of said
54 integrity constraints are not satisfied; and

55 after a valid change of state of an object acted upon by
56 said object server occurs, test trigger relationships or condition-
57 action rules specified in said Formal Language Specification and, if
58 any trigger event is satisfied, triggering a service specified in said
59 condition-action rule or trigger relationship.

1 11. [Cancelled]

1 12. [Cancelled]

1 13. [Previously Presented] A process for converting a Formal Language

2 Specification encoding a Conceptual Model which defines desired system logic and a
3 desired user interface of a desired computer program into source code which encodes
4 said desired computer program, comprising:

5 validating said Formal Language Specification to ensure it is complete and correct;
6 retrieving predetermined information from said Formal Language Specification
7 encoding a Conceptual Model which defines one or more classes of objects in an object
8 model, a dynamic model which specifies the behavior of each object in response to
9 services, triggers and global transactions as represented by a state transition diagram
10 for every class and an object interaction diagram for every trigger and for every global
11 transaction, and a functional model which defines the semantics of any change of each
12 object's state as a consequence of an event occurrence by specifying for each class
13 one or more mathematical or logical formulas which define how one or more variable
14 attributes of said class will have their values changed when one or more specified
15 events of said class occurs meaning one or more services of said class is executed;

16 using predetermined information retrieved from said Formal Language
17 Specification to automatically generate source code which implements a presentation tier
18 of said desired computer program;

19 using predetermined information retrieved from said Formal Language
20 Specification to automatically generate source code which implements a persistence tier,
21 database of data structure of said desired computer program;

22 using said predetermined information retrieved from said Formal Language
23 Specification to automatically generate source code which implements a middle tier of
24 said desired program which communicates with said presentation tier in the manner
25 defined below, said middle tier written by automatically generating at least the following
26 component instances for each class defined in said Formal Language Specification:

27 a server component instance including a method to implement each
28 service present in a signature of said class and one or more methods to
29 receive requests from said presentation tier that relate to execution of
30 services of said class;

31 a query component instance including a method for implementing
32 queries to extract information from said persistence tier relating to objects
33 within said class and a method to receive and process requests from said
34 presentation tier to query said persistence tier;

an executive component instance including one or more methods to receive and process a request from said server component or another executive component to execute a service in said class and carry out the following functions:

verify the existence and validity for a requested server component;

create a copy of a requested server component instance in memory and access said persistence tier using said query component to retrieve values of constant and variable attributes of said server component;

validate state transitions for said requested service and a present state for a server component instance;

verify the satisfaction of preconditions specified in said
Formal Language Specification of said requested service;

changing the state of said server component instance to a new state by modifying a value of a variable attribute of said server component instance by performing all valuations specified in said functional model affected by said requested service;

validating said new state by verifying said new state does not violate static or dynamic restrictions specified in said Formal Language Specification;

check trigger conditions established in said Formal Language Specification to determine if said new state causes any trigger to occur, and, if so, which actions should be carried out;

communicate with said persistence tier to access or store all attributes of said server component instance.

14. [currently amended] A process for validating a ~~formal language specification~~Formal Language Specification, comprising the steps:

A) checking said formal language specificationFormal Language Specification to ensure that it is complete in that all required properties of a Conceptual Model embodied in said formal language specificationFormal Language Specification are defined and have a valid value;

7 B) using rules of grammar of whatever formal language said formal
8 language specification Formal Language Specification is written in, checking said
9 formal language specification Formal Language Specification to ensure it is correct
10 in that it is syntactically and semantically correct and not ambiguous.

1 15. [Previously Presented] The process of claim 14 further comprising the step
2 of presenting a request for said missing information via a mechanism of a user interface
3 if any information is missing or presenting a request via a user interface mechanism to
4 correct any syntactic or semantic error or clarify any ambiguity discovered during said
5 validation process.

1 16. [Currently Amended] The process of claim 14 further comprising the step of
2 doing a partial validation each time an element is added to said ~~formal language~~
3 specification Formal Language Specification to check for completeness and correctness
4 and mark the portion of said ~~formal language specification~~ Formal Language Specification
5 just added as invalid if an error is found so that a request to correct said error is can be
6 presented later when a full validation of said ~~formal language specification~~ Formal
7 Language Specification is requested.

1 17. [Previously Presented] The process of claim 14 wherein step A comprises
2 checking to ensure that all elements in said Conceptual Model have a set of properties
3 that exist and have a valid value and wherein step B comprises using a predetermined
4 process and grammar for every type of formula in said Conceptual Model to ensure each
5 is syntactically and semantically correct.

1 18. [Previously Presented] The process of claim 14 wherein step A comprises
2 performing strict validation on some element properties and flexible validation of other
3 element properties of said Conceptual Model, said strict validation of a property defined
4 as requiring a full definition and valid value for a property, and flexible validation defined
5 as allowing a property to be incomplete or have an invalid value during a process of
6 inputting elements which define said Conceptual Model, and wherein the following table
7 defines which elements and properties have strict or flexible validations:

| Element | Property | Subproperty | Validation Type |
|-----------|--|----------------|-----------------|
| class | | | |
| | name | | strict |
| | ID function | | flexible |
| | attributes (at least one) | | flexible |
| | services (at least a Create service) | | flexible |
| | static and dynamic integrity constraints | their formulas | strict |
| attribute | | | |
| | name | | strict |
| | type (constant, variable, derived) | | strict |
| | data-type (real, integer, etc.) | | strict |
| | default value | | strict |
| | size | | strict |
| | request in creation service | | strict |
| | null value allowed | | strict |
| | evaluations (variable attributes) | | flexible |
| | derivation formula (derived attributes) | | flexible |

| | | | |
|-------------------------------|--------------------------------------|-----------------------------|----------|
| Evaluation | | | |
| | one variable attribute of a class | | strict |
| | one service of the same class | | strict |
| | condition | | strict |
| | formula of evaluation | | strict |
| derivation | | | |
| | formula | | strict |
| | condition (optional) | | strict |
| service | | | |
| | name | | strict |
| | arguments | | |
| | | argument's name | strict |
| | | data type | strict |
| | | default value (optional) | strict |
| | | null value | strict |
| | | size (if proceeds) | strict |
| | | formula of transaction | flexible |
| preconditions of an action | | | |
| | formula | | strict |

| | | agents affected by condition | strict |
|------------------------------|--|------------------------------|--------|
| relationship: aggregation | | | |
| | related classes (component and composite) | | strict |
| | relationship name | | strict |
| | both directions: role names | | strict |
| | cardinality | | strict |
| | inclusive or referential | | strict |
| | dynamic | | strict |
| | clause "group by" (optional) | | strict |
| | insertion and deletion events (if proceed) | | strict |
| relationship: inheritance | | | |
| | related classes (parent & child) | | strict |
| | temporal (versus permanent) | | strict |
| | specialization condition or events | | strict |

| | | | |
|--|---|------------------------------|----------|
| relationship: agent | | | |
| | agent class and service allowed to activate | | strict |
| state transition diagram | | | |
| | all states of classes (three at least) | | flexible |
| state in state transition diagram | | | |
| | name | | strict |
| transition in state transition diagram | | | |
| | estate of origin | | strict |
| | estate of destination | | strict |
| | service of class | | strict |
| | | control condition (optional) | strict |
| trigger | | | |
| | condition | | strict |
| | class or instance of destination | | strict |
| | target (self, object, class) | | strict |
| | activated service | | strict |

| | | | |
|---------------------|---|-----------------|---------|
| | service arguments' initialization (optional) | | |
| | | argument values | strict |
| global interactions | | | |
| | name | | strict |
| | formula | | strict |
| user exit functions | | | |
| | name | | strict |
| | return data-type | | strict |
| | arguments, (optional) | | |
| | argument's name | | strict |
| | argument's data-type | | strict. |

1 19. [Previously Presented] The process of claim 14 wherein step B comprises
 2 validating formulas to ensure each formula complies with a precise syntax defined for a
 3 formula of that type and is semantically correct, where there are several types of
 4 formulas, and wherein a predetermined process for validation and a set of rules of
 5 grammar exist for each type of formula and a validation process and a set of rules
 6 appropriate for the type of formula being validated is used for validation of each formula.

1 20. [Currently Amended] The process of claim 14 wherein step B comprises
 2 validating formulas to ensure each formula complies with a precise syntax defined for a
 3 formula of that type and is semantically correct, where there are several types of
 4 formulas defined by the table below:

| | |
|--|--|
| default value calculation of | |
| | class attributes (constant and variable) |
| | service and transaction arguments |
| inheritance: specialization conditions | |
| static and dynamic integrity constraints | |
| derivations and valuations | |
| | effect formula (derived or variable attributes respectively) |
| | conditions (optional) |
| preconditions for actions | |
| control conditions for transitions in state transition diagram | |
| triggering conditions | |
| local and global transactions formulas | |

1 and further comprising the steps of:

2 presenting dialog boxes via a user interface by which a user is provided a
 3 mechanism to may enter said formulas during a process of defining said
 4 Conceptual Model;
 5 and wherein step B is performed by preventing a user from leaving a dialog box being
 6 used to define a formula until said formula being defined is syntactically and semantically
 7 correct.

1 21. [Previously Presented] The process of claim 14 wherein steps A and B are
 2 performed by at least checking to ensure that every formula is syntactically correct,
 3 every class has an identification function and has a creation event and a destroy event,
 4 every triggering formula is semantically correct, every name of an aggregation is unique
 5 in the scheme of said Conceptual Model, every derived attribute has at least a derivation
 6 formula, every service has an agent or server declared to execute it.

1 22. [Currently Amended] The process of claim 14 further comprising performing
2 steps A and B each time a user working on defining said Conceptual Model makes a
3 change which may invalidate one or more formulas, but wherein predetermined formulas
4 are allowed to be temporarily incorrect so that the user is presented them for can review
5 them at a later time if they are still incorrect when a full validation process is performed
6 after the user is done defining the Conceptual Model.

1 23. [Currently Amended] The process of claim 14 further comprising the step of
2 using a computer to automatically translate said formal language specificationFormal
3 Language Specification into computer code after steps A and B and been completed and
4 all formulas are syntactically and semantically complete and correct.

1 24. [Currently Amended] The process of claim 14 further comprising the steps:
2 presenting user interface tools by which a user may define said
3 Conceptual Model and make changes thereto;
4 checking all affected formulas each time a change is made to said
5 Conceptual Model;
6 if the change affects a strictly validated property, then the change is
7 rejected if the property is not given a valid value, otherwise the change is
8 accepted;
9 if the change affects a property which is not strictly validated, then the
10 user is informed should any error arise, but allowed to do the modification if said
11 user he or she wishes;
12 if there are no affected formulas, modifying the Conceptual Model as
13 specified by the user.

1 25. [Currently Amended] The process of claim 14 wherein user interface defining
2 portions of said formal language specificationFormal Language Specification are
3 validated by:
4 verifying that any patterns defined by said user are acceptable user
5 interface patterns with no essential information missing;
6 attributes used in filters specified as part of said user interface are visible

7 from a definition class;
8 attributes used in order criteria are visible from a definition class;
9 any formula in a filter is syntactically and semantically correct and uses
10 only terms defined in said Conceptual Model;
11 any action selection pattern uses as final actions object defined in said
12 Conceptual Model;
13 any set of dependency patterns are terminal and have confluence; and
14 warnings are displayed to said user if any pattern is defined but not used or if an
15 instance pattern is duplicated.

1 26. [Currently Amended] An apparatus comprising a computer programmed with
2 an operating system and a validation program that cooperates with said operating system
3 to control said computer to perform the following functions of a validation process:

4 A) check a ~~formal language specification~~Formal Language Specification
5 for completeness by checking to ensure said ~~formal language specification~~Formal
6 Language Specification has no missing information which is needed to detail the
7 requirements of a desired computer program modelled by said ~~formal language~~
8 specificationFormal Language Specification;
9 Bb) cooperate with said operating system to ensure said ~~formal language~~Formal Language Specification
10 is correct by checking each
11 statement in said ~~formal language specification~~Formal Language Specification
12 according to rules of grammar of whatever formal language in which said ~~formal~~
13 language specificationFormal Language Specification is written to ensure that
14 each statement is syntactically and semantically correct and not ambiguous so as
15 to ensure that all properties of elements in a Conceptual Model encoded in said
16 ~~formal language specification~~Formal Language Specification have a value which
17 is valid and to ensure that all formulas in said Conceptual Model have correct
18 syntax and meaning.

1 27. [Previously Presented] The apparatus of claim 26 wherein said validation
2 program controls said computer to perform the following additional function:
3 presenting a request for said missing information via a mechanism of a
4 user interface if any information is missing or presenting a request via a user

5 interface mechanism to correct any syntactic or semantic error or clarify any
6 ambiguity discovered during said validation process.

1 28. [Currently Amended] The apparatus of claim 26 wherein said validation
2 program controls said computer to perform the following additional function:

3 doing a partial validation each time an element is added to said formal
4 language specification Formal Language Specification to check for completeness
5 and correctness and mark the portion of said formal language specification Formal
6 Language Specification just added as invalid if an error is found so that a request
7 to correct said error is can be presented later when a full validation of said formal
8 language specification Formal Language Specification is requested if said error
9 still exists.

1 29. [Previously Presented] The apparatus of claim 26 wherein said validation
2 program controls said computer to perform the following additional function:

1 30. [Previously Presented] The apparatus of claim 26 wherein said validation
2 program controls said computer to perform step B by validating formulas to ensure each
3 formula complies with a precise syntax defined for a formula of that type and is
4 semantically correct, where there are several types of formulas, and wherein a
5 predetermined process for validation and a set of rules of grammar exist for each type of
6 formula and a validation process and a set of rules appropriate for the type of formula
7 being validated is used for validation of each formula.

1 31. [Previously Presented] The apparatus of claim 26 wherein said validation
2 program controls said computer to perform step B by validating formulas to ensure each
3 formula complies with a precise syntax defined for a formula of that type and is
4 semantically correct, where there are several types of formulas defined by the table

5 below:

| | |
|--|--|
| default value calculation of | |
| | class attributes (constant and variable) |
| | service and transaction arguments |
| inheritance: specialization conditions | |
| static and dynamic integrity constraints | |
| derivations and valuations | |
| | effect formula (derived or variable attributes respectively) |
| | conditions (optional) |
| preconditions for actions | |
| control conditions for transitions in state transition diagram | |
| triggering conditions | |
| local and global transactions formulas | |

6 and wherein said validation program is further structured to control said computer to
7 perform the steps of:

8 presenting dialog boxes via a user interface by which a user may enter
9 said formulas during a process of defining said Conceptual Model;
10 and wherein step B is performed by preventing a user from leaving a dialog box being
11 used to define a formula until said formula being defined is syntactically and semantically
12 correct.

1 32. [Currently Amended] The apparatus of claim 26 wherein said validation
2 program controls said computer to perform the following steps:
3 presenting user interface tools by which a user may define said
4 Conceptual Model and make changes thereto;
5 checking all affected formulas each time a change is made to said

1 33. [Currently Amended] An physical computer readable storage media medium
2 having stored thereon computer-readable instructions which when executed by a
3 computer implement a validation program to control said computer to perform the
4 following functions of a validation process:

5 A) check a ~~formal language specification~~Formal Language Specification
6 for completeness by checking to ensure said ~~formal language specification~~Formal
7 Language Specification has no missing information which is needed to detail the
8 requirements of a desired computer program modelled by said ~~formal language~~
9 ~~specification~~Formal Language Specification;

10 Bb) cooperate with said operating system to ensure said formal language
11 specification Formal Language Specification is correct by checking each
12 statement in said formal language specification Formal Language Specification
13 according to rules of grammar of whatever formal language in which said formal
14 language specification Formal Language Specification is written to ensure that
15 each statement is syntactically and semantically correct and not ambiguous so as
16 to ensure that all properties of elements in a Conceptual Model encoded in said
17 formal language specification Formal Language Specification have a value which
18 is valid and to ensure that all formulas in said Conceptual Model have correct
19 syntax and meaning.

1 34. [Previously Presented] The computer-readable medium of claim 33 wherein
2 said validation program is further structured to control said computer to perform the
3 following additional function:

4 presenting a request for said missing information via a mechanism of a
5 user interface if any information is missing or presenting a request via a user
6 interface mechanism to correct any syntactic or semantic error or clarify any
7 ambiguity discovered during said validation process.

1 35. [Currently Amended] The computer-readable medium of claim 33 wherein said
2 validation program is further structured to control said computer to perform the following
3 additional function:

4 doing a partial validation each time an element is added to said formal
5 language specification Formal Language Specification to check for completeness
6 and correctness and mark the portion of said formal language specification Formal
7 Language Specification just added as invalid if an error is found so that a request
8 to correct said error is ~~can be~~ presented later when a full validation of said formal
9 language specification Formal Language Specification is requested if said error
10 still exists.

1 36. [Previously Presented] The computer-readable medium of claim 33 wherein
2 said validation program is further structured to control said computer to perform the
3 following additional functions:

1 37. [Previously Presented] The computer-readable medium of claim 29 wherein
2 said validation program is structured to control said computer to perform step B by
3 validating formulas to ensure each formula complies with a precise syntax defined for a
4 formula of that type and is semantically correct, where there are several types of
5 formulas, and wherein a predetermined process for validation and a set of rules of
6 grammar exist for each type of formula and a validation process and a set of rules
7 appropriate for the type of formula being validated is used for validation of each formula.

1 38. [Previously Presented] The computer-readable medium of claim 33 wherein
 2 said validation program is structured to control said computer to perform step B by
 3 validating formulas to ensure each formula complies with a precise syntax defined for a
 4 formula of that type and is semantically correct, where there are several types of
 5 formulas defined by the table below:

| | |
|--|--|
| default value calculation of | |
| | class attributes (constant and variable) |
| | service and transaction arguments |
| inheritance: specialization conditions | |
| static and dynamic integrity constraints | |
| derivations and valuations | |
| | effect formula (derived or variable attributes respectively) |
| | conditions (optional) |
| preconditions for actions | |
| control conditions for transitions in state transition diagram | |
| triggering conditions | |
| local and global transactions formulas | |

6 and wherein said validation program is further structured to control said computer to
 7 perform the steps of:

8 presenting dialog boxes via a user interface by which a user may enter
 9 said formulas during a process of defining said Conceptual Model;
 10 and wherein step B is performed by preventing a user from leaving a dialog box being
 11 used to define a formula until said formula being defined is syntactically and semantically
 12 correct.

1 39. [Currently Amended] The computer-readable medium of claim 33 wherein said

2 validation program is structured to control said computer to perform the following steps:
3 presenting user interface tools by which a user may define said
4 Conceptual Model and make changes thereto;
5 checking all affected formulas each time a change is made to said
6 Conceptual Model;
7 if the change affects a strictly validated property, then the change is
8 rejected if the property is not given a valid value, otherwise the change is
9 accepted;
10 if the change affects a property which is not strictly validated, then the
11 user is informed should any error arise, but allowed to do the modification if said
12 user he or she wishes;
13 if there are no affected formulas, modifying the Conceptual Model as
14 specified by the user.

1 40. [New] A process to automatically translate a Formal Language Specification
2 defining the functionality of a computer application modelled in a Conceptual Model, into a
3 computer program called an application, said process comprising the steps of:

4 A) validating said Formal Language Specification to ensure said Formal
5 Language Specification is complete in that there is no missing information in said
6 Formal Language Specification and to ensure said Formal Language Specification
7 is correct in that primitives of said conceptual model are syntactically and
8 semantically consistent and not ambiguous;

9 B) translating said validated Formal Language Specification into computer
10 readable code which has the capability to control a computer to provide a user
11 interface access mechanism to allow users to log in by entering at least
12 identification data and to use said identification data to authenticate and validate a
13 user as an instance of a class of the validated fFormal Language Specification
14 that act as agent in at least one agent relationship;

15 C) translating said validated Formal Language Specification into computer
16 readable code which has the capability to control a computer to provide a view of
17 the system defining the set of objects and attributes the user can query and the
18 set of services said user can execute, the content of said system view

19 depending on the identity of said user accessing said application;
20 and

21 D) translating said validated Formal Language Specification into computer
22 readable code which has the capability to control a computer to provide user
23 interface interaction mechanisms to interact with and execute the functionality of
24 the application in terms of performing queries on information managed by said
25 application and executing services to modify the state of said information
26 managed by said application, said services comprising events, local transactions
27 and global transactions.

1 41. [New] The process of claim 40 wherein Step D comprises translating said
2 Formal Language Specification into computer code which has the capability to control a
3 computer so as to implement functionality of said queries as defined in said validated
4 Formal Language Specification by means of filter formulas of filter patterns representing
5 said queries, and so as to implement functionality of said services as defined in said
6 validated Formal Language Specification in terms of:

- 7 - at least a valuation for each variable attribute of each class in said validated
8 Formal Language Specification associated to at least an event of said class,
9 which altogether define the functionality of every event;
- 10 - a transaction formula that defines a composition of services into a molecular
11 execution unit, thereby defining functionality of every local transaction and global
12 transaction;
- 13 - state transitions to control the valid lives for objects of each class in said
14 validated Formal Language Specification, upon occurrence of an event or a local
15 transaction;
- 16 - optional preconditions to the execution of services; and
- 17 - optional integrity constraints to prevent the execution of services from leaving the
18 information managed by said application in an inconsistent or invalid state.

1 42. [New] The process of claim 40 wherein said step B creates computer
2 readable code which has the capability to control a computer to provide user interface
3 access mechanisms which block access such that only users that belong to any class
4 of said view for which said computer application is produced can connect or log on to

5 said desired computer program, said control being performed by requesting that a user
6 wishing to log onto said desired computer program indicate information that identifies a
7 class of which said user is an instance, and also indicate information that identifies the
8 user as an instance of said class so as to identify said user, and also indicate
9 information that is used as a password for that user so as to authenticate said user.

1 43. [New] The process of claim 40 wherein said step C creates computer
2 readable code which has the capability to control a computer to provide said view of the
3 system in such a manner that said system view comprises user interaction mechanisms
4 included in an Action Selection Presentation Pattern associated with said view for which
5 the application is produced.

1 44. [New] The process of claim 43 wherein step C creates computer readable
2 code which has the capability to control a computer to restrict the user interface
3 interaction mechanisms to the ones the user who logged on is allowed to interact with
4 according to privileges established by data structures which relate the class said user
5 belongs to, playing the role of agent, with classes, playing the role of servers, that
6 determine which services of each server class will be available for execution by said
7 user and which attributes of each server class said user will be able to query.

1 45. [New] The process of claim 40 wherein step D comprises translating said
2 validated Formal Language Specification into computer readable code which has the
3 capability to control a computer to provide user interface interaction mechanisms which
4 allow a user who has logged on to interact with and access functionality of said
5 computer application by invoking services, executing queries and/or execution of user
6 interface interaction scenarios.

1 46. [New] The process of claim 45 wherein step D comprises the steps of
2 translating said Formal Language Specification into computer readable code which has
3 the capability to control a computer to provide user interface interaction mechanisms
4 which allow a user who has logged in to interact with and access the functionality of
5 said computer application by execution of only selected services selected from a group
6 of services comprising events, local transactions and/or global transactions and

7 depending upon the identity of said user.

1 47. [New] The process of claim 46 where the steps in step D of translating said
2 Formal Language Specification to generate computer readable code which controls a
3 computer to allow a logged on user to interact with and execute the functionality
4 associated to an event comprise generating code which controls a computer to provide a
5 mechanism to construct the message associated to said event and to execute said
6 event.

1 48. [New] The process of claim 47 wherein said step D includes the steps of
2 translating said validated Formal Language Specification to generate computer readable
3 code which controls a computer to provide a mechanism to construct said message
4 associated to said event comprises generating code to control a computer to perform the
5 following steps:

6 - providing a mechanism to identify the object on which the event will be
7 executed, except if said event is a creation event, in which case this
8 step will be omitted;
9 - providing mechanisms to give a value to every argument of said event,
10 said mechanisms further controlling that arguments that require a value
11 are provided a value and that any argument whose value is provided
12 receives a valid value.

1 49. [New] The process of claim 47 wherein said step D includes the steps of
2 translating said validated Formal Language Specification to generate computer readable
3 code which has the capability to control a computer to provide a mechanism to execute a
4 service which is an event and comprises the following steps:

5 - translating said validated Formal Language Specification so as to provide
6 code which controls a computer to provide a mechanism to, except in the
7 case of a creation event, verify the existence of said object on which
8 said event will be executed, or the non-existence of the object to be
9 created in the case of a creation event;
10 - translating said validated Formal Language Specification so as to provide
11 code which controls a computer to provide a mechanism to, except in the

case of a creation event, recover the state of the object on which the event is executed from whatever memory or database or repository or any other persistence means (hereafter just "memory") to which said state of said object has been saved;

- translating said validated Formal Language Specification so as to provide code which controls a computer to provide a mechanism to verify that, according to the state transition diagram of the class owning said event, there is a valid state transition labelled with said event being executed and for the agent class to which the user belongs who logged onto said desired computer program, in which case said mechanism will update the state of the object on which the event is executed according to said state transition diagram, and if there is no valid state transition, said mechanism will produce an error message causing the execution of the event to stop and roll back all changes made to the state of the object on which said event is executed

- translating said validated Formal Language Specification so as to provide code which controls a computer to provide a mechanism to verify that every precondition that is defined for said event being executed and for said agent class to which the user logged on to the computer application belongs, is satisfied, and should any of said preconditions not be satisfied, then said mechanism will produce an error with the error message being predefined for said precondition that does not hold causing the execution of the event to stop and roll back all changes made to the state of said object on which said event is executed;

- except in the case of creation events or destruction events, translating said validated Formal Language Specification so as to provide code which controls a computer to provide a mechanism to produce the changes of values to the variable attributes of said class owning said event for which valuation formulas in said functional model have been defined such that said valuations relate said variable attributes with the event being executed, and wherein said mechanism applies only the changes to the variable attributes which are required by valuation formulas of valuations whose valuation condition formula evaluates to

45 true, if any, or the change required by a valuation formula of a valuation
46 having no valuation condition formula;
47 - in the case of a creation event, translating said validated Formal
48 Language Specification so as to provide code which has the capability to
49 control a computer to provide a mechanism to assign a value to every
50 constant or variable attribute of an object on which said creation event is
51 executed and establishing relationships between said object on which
52 said creation event is executed with objects of classes related with the
53 class owning said creation event;
54 - in the case of a destruction event, translating said validated Formal
55 Language Specification so as to provide code which has the capability to
56 control a computer to provide a mechanism to delete relationships of the
57 object on which said destruction event is executed with objects said
58 object is related to;
59 - except in the case of destruction events, translating said validated
60 Formal Language Specification so as to provide code which has the
61 capability to control a computer to provide a mechanism to check that
62 every integrity constraint defined in the class owning said event is
63 satisfied, for the object whose state has been changed by said event,
64 and should any of said integrity constraints not be satisfied, said
65 mechanism will produce an error with the error message predefined for
66 said integrity constraint that does not hold, said mechanism causing the
67 execution of the event to stop and roll back all changes made to the state
68 of the object on which the event is executed;
69 - except in the case of a destruction event, translating said validated
70 Formal Language Specification so as to provide code which has the
71 capability to control a computer to provide a mechanism to save the
72 changes made to said object on which the event is executed to memory;
73 - in the case of a destruction event, translating said validated Formal
74 Language Specification so as to provide code which has the capability to
75 control a computer to provide a mechanism to delete the object on which
76 said destruction event is executed from memory to which said object has
77 been saved;

78 - except in the case of a destruction event, translating said validated
79 Formal Language Specification so as to provide code which has the
80 capability to control a computer to provide a mechanism to check every
81 trigger condition on said object on which said event is executed for
82 every trigger relationship defined on said class owning said event, and
83 wherein for every trigger condition that holds, a mechanism to execute
84 the service associated to said trigger relationship by:

- providing a mechanism to determine the set of objects on which said service associated to said trigger will be executed, and
- executing said service on every object of said set of objects by:
 - giving a value to every argument of said service, and ensuring that every argument that requires a value is provided a value and that any argument whose value is provided receives a valid value; and
 - invoking the execution of said service on said object;

95 and
96 - translating said validated Formal Language Specification so as to provide
97 code which has the capability to control a computer to provide a
98 mechanism to inform the requestor of the result of executing the event.

1 50. [NEW] The process of claim 46 where the steps in step D of translating said
2 Formal Language Specification to generate computer readable code which controls a
3 computer to allow a logged on user to interact with and execute the functionality
4 associated to a local transaction comprise generating code which controls a computer to
5 provide a mechanism to construct the message associated to said local transaction and
6 to execute said local transaction.

1 51. [New] The process of claim 50 wherein the steps in step D of translating said
2 Formal Language Specification to generate computer readable code which controls a
3 computer to provide a mechanism to construct said message associated to the local
4 transaction comprise generating code to control a computer to perform the following

5 steps:

6 - providing a mechanism to identify the object on which said local
7 transaction will be primarily executed, except if said transaction is a
8 creation service, in which case this step will be omitted;
9 - providing mechanisms to give a value to every argument of said local
10 transaction, said mechanisms further controlling that arguments that
11 require a value are provided a value and that any argument whose value
12 is provided receives a valid value.

1 52. [New] The process of claim 50 wherein said step D includes the steps of
2 translating said validated Formal Language Specification to generate computer readable
3 code which has the capability to control a computer to execute a service which is a local
4 transaction and comprises the following steps:

5 - translating said validated Formal Language Specification so as to provide code
6 which controls a computer to provide a mechanism to, except in the case of a
7 creation service, verify the existence of the object on which the local
8 transaction will be executed or, in the case of a creation service, verify the
9 non-existence of the object to be created;
10 - translating said validated Formal Language Specification so as to provide code
11 which controls a computer to, except in the case of a creation service,
12 provide a mechanism to recover the state of the object on which the local
13 transaction is executed from memory to which the state of said object has
14 been saved;
15 - translating said validated Formal Language Specification so as to provide code
16 which controls a computer to provide a mechanism to verify that, according to
17 the state transition diagram of the class owning said local transaction, there is
18 a valid state transition labelled with said local transaction being executed and
19 for the agent class to which the user logged on to the system belongs, and, if
20 there is such a valid state transition, said mechanism will update the state of
21 the object on which the local transaction is executed according to said state
22 transition diagram, and, if there is no such valid state transition, said
23 mechanism will produce an error message causing the execution of said local
24 transaction to stop and roll back all changes made to the state of said object

25 on which the local transaction is executed and of any other object the state of
26 which has been modified by the execution of said local transaction;
27 translating said validated Formal Language Specification so as to provide code
28 which controls a computer to provide a mechanism to verify that every
29 precondition that is defined for the local transaction being executed and for
30 the agent class to which the user logged on to the system belongs, is
31 satisfied, and should any of said preconditions not hold, then said mechanism
32 will produce an error with the error message defined for said precondition
33 that does not hold causing the execution of said local transaction to stop and
34 roll back all changes made to the state of the object on which said local
35 transaction is executed and of any other object the state of which has been
36 modified by the execution of said local transaction;
37 for every service comprised in the transaction formula of said local
38 transaction being executed:
39 o if said service has an associated guard, translating said validated
40 Formal Language Specification so as to provide code which controls a
41 computer to provide a mechanism to check that said guard associated
42 to said service holds; and if said guard does not hold, the rest of the
43 steps associated to said service will be omitted;
44 o translating said validated Formal Language Specification so as to
45 provide code which controls a computer to provide a mechanism to
46 determine the set of objects on which said service comprised in said
47 transaction formula will be executed and to provide a mechanism to
48 execute said service on every object of said set of objects by:
49 • providing mechanisms to give a value to every argument of said
50 service, said mechanisms further ensuring that arguments that
51 require a value are provided a value and that any argument whose
52 value is provided receives a valid value;
53 • providing a mechanism to invoke the execution of said service on
54 said object and control the result of said execution, and should said
55 execution result in an error, causing the execution of the local
56 transaction to stop and roll back all changes made to the state of
57 said object on which the local transaction is executed and of any

58 other object the state of which has been modified by the execution
59 of said local transaction

60 - translating said validated Formal Language Specification so as to provide code
61 which has the capability to control a computer to provide a mechanism to
62 check that every integrity constraint holds which is defined in the class
63 owning the local transaction and in classes whose instances include objects
64 the state of which has been modified by the execution of said local
65 transaction, for any object whose state has been changed by said local
66 transaction, and, should any of said integrity constraints not hold, said
67 mechanism produces an error with an error message defined for said integrity
68 constraint that does not hold and causing the execution of said local
69 transaction to stop and roll back all changes made to the state of the object on
70 which the local transaction is executed and of any other object the state of
71 which has been modified by the execution of said local transaction;
72 - translating said validated Formal Language Specification so as to provide code
73 which has the capability to control a computer to provide a mechanism to
74 check every trigger condition on any object the state of which has been
75 modified by the execution of said local transaction, for every trigger
76 relationship defined on each class owning an object the state of which has
77 been changed by execution of said local transaction, and for every trigger
78 condition that holds, a mechanism to execute the service associated to said
79 trigger relationship by:

- providing a mechanism to determine the set of objects on which said service associated to said trigger will be executed, and
- providing a mechanism to execute said service on every object of said set of objects by:
 - giving a value to every argument of said service, and ensuring that every argument that requires a value is provided a value and that any argument whose value is provided receives a valid value; and
 - invoking the execution of said service on said object;

89 and

90 - translating said validated Formal Language Specification so as to provide code

91 which has the capability to control a computer to provide a mechanism so as
92 to inform the requestor of the result of executing said local transaction.

1 53. [New] The process of claim 46 wherein the steps in step D of translating said
2 Formal Language Specification to generate computer readable code which controls a
3 computer to allow a logged on user to interact with and execute the functionality
4 associated to a global transaction comprise generating code which controls a computer
5 to provide a mechanism to construct the message associated to said global transaction
6 and to execute said global transaction.

1 54. [New] The process of claim 53 wherein the steps in step D of translating said
2 Formal Language Specification to generate computer readable code which controls a
3 computer to provide a mechanism to construct said message associated to said global
4 transaction comprises the step of generating code which controls a computer to provide
5 mechanisms to give a value to every argument of the global transaction, said
6 mechanisms further controlling that arguments that require a value are provided a value
7 and that any argument whose value is provided receives a valid value.

1 55. [New] The process of claim 53 wherein said step D includes the steps of
2 translating said validated formal language specification to generate computer readable
3 code which can control a computer to execute a service which is a global transaction
4 and comprises the following steps:

5 - translating said validated formal language specification so as to provide code
6 which controls a computer to provide a mechanism to verify that every
7 precondition that is defined for the global transaction being executed and for
8 the agent class the user logged on to the system belongs to, is satisfied, and
9 should any of said preconditions not hold, then said mechanism will produce
10 an error with the error message being defined for said precondition that does
11 not hold, and causing execution of said global transaction to stop and roll back
12 all changes made to any object the state of which has been modified by
13 execution of said global transaction;
14 - for every service included in a transaction formula of said global transaction
15 being executed:

16 o if said service has an associated guard, translating said validated formal
17 language specification so as to provide code which controls a computer to
18 provide a mechanism to check that said guard associated to said service
19 holds, and if said guard does not hold, the rest of the steps associated to
20 said service will be omitted;

21 o translating said validated formal language specification so as to provide
22 code which controls a computer to provide a mechanism to determine the
23 set of objects on which said service will be executed and to provide a
24 mechanism to execute said service on every object of said set of objects
25 by:

26 • providing mechanisms to give a value to every argument of said
27 service, said mechanisms further ensuring that arguments that
28 require a value are provided a value and that any argument whose
29 value is provided receives a valid value;

30 • providing a mechanism to invoke the execution of said service on
31 said object and control the result of said execution, and should said
32 execution result in an error, causing the execution of the global
33 transaction to stop and roll back all changes made to the state of
34 any object the state of which might have been modified by the
35 execution of said global transaction;

36 - translating said validated formal language specification so as to provide code
37 which can control a computer to provide a mechanism to check that every
38 integrity constraint holds that is defined in classes whose instances include
39 objects the state of which has been modified by the execution of said global
40 transaction, for any object whose state has been changed by said global
41 transaction, and should any of said integrity constraints not hold, said
42 mechanism produces an error with an error message defined for said integrity
43 constraint that does not hold thereby causing execution of said global
44 transaction to stop and roll back all changes made to any object the state of
45 which has been modified by execution of said global transaction;

46 - translating said validated formal language specification so as to provide code
47 which can control a computer to provide a mechanism to check every trigger
48 condition on any object the state of which has been modified by execution of

49 said global transaction, for every trigger relationship defined on each class
50 owning any object the state of which has changed by execution of said global
51 transaction, and, for every trigger condition that holds, a mechanism to
52 execute the service associated to said trigger relationship by:
53 • providing a mechanism to determine a set of objects on which said
54 service associated to said trigger will be executed, and
55 • providing a mechanism to execute said service on every object of
56 said set of objects by:
57 • giving a value to every argument of said service, and
58 ensuring that every argument that requires a value is
59 provided a value and that any argument whose value is
60 provided receives a valid value; and
61 • invoking the execution of said service on said object;
62 and
63 - translating said validated formal language specification so as to provide code
64 which can control a computer to provide a mechanism so as to inform the
65 requestor of the result of executing said global transaction.

1 56. [New] The process of claim 45 wherein the steps in step D of translating said
2 formal language specification to generate computer readable code which can control a
3 computer to allow a logged on user to interact with and execute the functionality of a
4 query as part of filter patterns independently or in the context of a Class Population
5 Presentation Pattern comprise generating computer code to control said computer to
6 provide a mechanism to construct the message associated with said filter and to execute
7 the query associated with said filter.

1 57. [NEW] The process of claim 56 wherein said step D includes the steps of
2 translating said validated formal language specification to generate computer readable
3 code which can control a computer to provide a mechanism to construct said message
4 associated to the filter comprise generating code to control a computer to perform the
5 following steps:
6 - providing a mechanism to indicate a filter whose associated query is to be
7 executed;

8 - providing mechanisms to give a value to every filter variable of a filter pattern
9 of said filter, said mechanisms further controlling that filter variables that
10 require a value are provided a value and that any filter variable whose value
11 is provided receives a valid value;
12 - providing a mechanism to indicate a display set pattern to be used in
13 performing the query
14 - providing a mechanism to indicate an order criterion pattern, if any, to be used
15 in performing said query.

1 58. [New] The process of claim 57 wherein said step D includes the steps of
2 translating said validated formal language specification to generate computer readable
3 code which can control a computer to provide a mechanism to execute the query
4 associated with said filter and comprises the following steps:

5 - translating said validated formal language specification so as to provide code
6 which controls a computer to provide a mechanism to access the population
7 of instances of the class owning said filter pattern from memory;
8 - translating said validated formal language specification so as to provide code
9 which controls a computer to provide a mechanism to retrieve instances of
10 said population which fulfil the condition stated by the filter formula of said
11 filter depending on the values assigned to every filter variable of said filter;
12 - translating said validated formal language specification so as to provide code
13 which controls a computer to provide a mechanism to access only part of the
14 state of every instance of said population matching said condition stated by
15 said filter formula of said filter, said part of said state being defined said
16 display set selected in said message, said part of said state dictated by said
17 display set being further constrained to the attributes said user logged onto
18 said desired computer program is allowed to query;
19 - translating said validated formal language specification so as to provide code
20 which controls a computer to provide a mechanism to return the instances of
21 said population which fulfil said condition stated by said filter formula and, if
22 an order criterion pattern has been indicated in said message, return said
23 instances of said population which fulfil said condition stated by said filter
24 formula in the order stated by said order criterion pattern.

1

1 59. [New] The process of claim 45 wherein step D comprises the steps of translating
2 said formal language specification into computer readable code which can control a
3 computer to provide user interface interaction mechanisms which allow a user who has
4 logged on to execute services and queries by interacting with one or more user interface
5 interaction scenarios implemented as Service Presentation Patterns, Instance
6 Presentation Patterns, Class Population Presentation Patterns and/or Master/Detail
7 Presentation Patterns.

1

1 60. [New] The process of claim 59 wherein the steps in step D of translating said
2 validated formal language specification to generate computer readable code which can
3 control a computer to allow a logged on user to execute a service by interaction with a
4 Service Presentation Pattern related to said service, said steps of translating comprising
5 steps to generate computer readable code which implements a set of mechanisms that
6 collaborate to construct a message for the execution of the service associated to said
7 Service Presentation Pattern, including:

8

- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to identify said Service Presentation Pattern by its alias;
- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to present a help message associated with said Service Presentation Pattern to said user interacting with said Service Presentation Pattern;
- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to identify each argument of said service associated with said Service Presentation Pattern by the alias of each said argument of said service;
- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to present each argument of said service associated with said Service Presentation Pattern to said user interacting with said Service Presentation Pattern in the order and groups dictated by the Arguments Grouping Presentation Pattern (if any) associated

24 to said Service Presentation Pattern;

25 - translating said validated formal language specification so as to provide code

26 which controls a computer to provide a mechanism to let the user provide a

27 value for each argument of the service associated to said Service

28 Presentation Pattern

29 - translating said validated formal language specification so as to provide code

30 which controls a computer to provide a mechanism to ensure that every

31 argument of said service associated with said Service Presentation Pattern

32 that requires a value has a value and to validate that every argument that has

33 a value has a valid value according to every said argument data type and said

34 Introduction Pattern, if any, associated with every said argument;

35 - translating said validated formal language specification so as to provide code

36 which controls a computer to provide a mechanism to present the user with

37 the default value, if any, of every argument of said service associated to said

38 Service Presentation Pattern;

39 - translating said validated formal language specification so as to provide code

40 which controls a computer to provide a mechanism to access an objects

41 selection mechanism corresponding to a Population Selection Pattern, if any,

42 associated with every object valuated argument of the service associated

43 with said Service Presentation Pattern;

44 - translating said validated formal language specification so as to provide code

45 which controls a computer to provide a mechanism to present said user with

46 a help message, if any, associated to every argument of said service

47 associated to said Service Presentation Pattern;

48 - translating said validated formal language specification so as to provide code

49 which controls a computer to provide a mechanism to implement a

50 Dependency Pattern, if any, associated with every argument of the service

51 associated with said Service Presentation Pattern, said mechanism:

52 o controlling a computer to monitor the occurrence of said user interface

53 interaction events relevant to every said argument, which either change the

54 value of said argument or activate/deactivate said argument;

55 o controlling a computer so as to check that the condition of an Event-

56 Condition-Action (hereafter ECA) rule of said Dependency Pattern holds,

57 and

58 o executing the actions in said ECA rule of said Dependency Pattern to assign

59 a value and/or activate and/or deactivate other arguments of the service

60 associated to said Service Presentation Pattern;

61 - translating said validated formal language specification so as to provide code

62 which controls a computer to provide a mechanism to present said user the

63 elements in a Display Set Pattern, if any, assigned as a Supplementary

64 Information Pattern to every object valued argument of said service

65 associated with said Service Presentation Pattern, whenever the value of

66 every said argument changes;

67 - translating said validated formal language specification so as to provide code

68 which controls a computer to provide a mechanism to allow a user to cancel

69 the interaction of said user with said interaction scenario represented by said

70 Service Presentation Pattern;

71 - a mechanism to confirm and send the message which will cause the service

72 associated to said Service Presentation Pattern to execute.

61. [New] The process of claim 59 wherein the steps in step D of translating said validated formal language specification to generate computer readable code which can control a computer to allow a logged on user query information on an instance of a class, execute services on said instance and/or navigate to interaction scenarios displaying information related with said instance, by interaction with an Instance Presentation Pattern of the class owning said instance, said steps of translating comprising the following steps:

8 - translating said validated formal language specification so as to provide code
9 which controls a computer to provide a mechanism to identify said Instance
10 Presentation Pattern by its alias;
11 - translating said validated formal language specification so as to provide code
12 which controls a computer to provide a mechanism to present a help
13 message, if any, associated with said Instance Presentation Pattern;
14 - translating said validated formal language specification so as to provide code
15 which controls a computer to provide a mechanism to present said user with
16 the value of each element in the Display Set Pattern of said Instance

Presentation Pattern, each element identified by the alias of said element and presented in the order dictated by said Display Set Pattern, said elements being further restricted to those elements that correspond to attributes the logged on user is allowed to query;

- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to access each user interface interaction scenarios corresponding to each service that can be executed on the object displayed by said Instance Presentation Pattern, each of said services identified by the alias of the Service Presentation Pattern corresponding to each of said services, said services being further restricted to the ones the logged on user is allowed to execute;
- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to present said user with a help message associated with each service, if any, executable on said object displayed by said Instance Presentation Pattern;
- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to access each user interface interaction scenario corresponding to each class, if any, owning instances related with the instance belonging to the class owning said Instance Presentation Pattern, each of said user interface interaction scenarios being identified by the alias of its corresponding Presentation Pattern, said user interface interaction scenarios being further restricted to the ones corresponding to classes said logged on user is allowed to query;
- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to present said user with any help message associated with each of the user interface interaction scenarios corresponding to each class, if any, owning instances related with instance belonging to the class owning said Instance Presentation Pattern;
- and
- translating said validated formal language specification so as to provide code which controls a computer to provide a mechanism to cancel the interaction of the logged on user with the user interface interaction scenario represented by said Instance Presentation Pattern.

50

1 62. [New] The process of claim 59 wherein the steps in step D of translating said
2 validated formal language specification to generate computer readable code which can
3 control a computer to allow a logged on user to query information on a set of instances of a
4 class, execute services on any instance in said set and or navigate to interaction scenarios
5 displaying information related with any said instance of said set, by interaction with a Class
6 Population Presentation Pattern of the class owning said set of instances, said steps of
7 translating comprising the following steps:

8 - translating said validated formal language specification so as to provide
9 code which controls a computer to provide a mechanism to identify said
10 Class Population Presentation Pattern by its alias;
11 - translating said validated formal language specification so as to provide
12 code which controls a computer to provide a mechanism to present any
13 help message associated with said Class Population Presentation Pattern;
14 - translating said validated formal language specification so as to provide
15 code which controls a computer to provide a set of mechanisms to obtain
16 and display a set of instances of the class owning said Class Population
17 Presentation Pattern, comprising:
18 o a mechanism to identify the Filter Pattern, if any, associated to said Class
19 Population Presentation Pattern;
20 o a mechanism to identify and select one of the Order Criterion Patterns, if
21 any, associated to said Class Population Presentation Pattern;
22 o a mechanism to identify each Filter Variable, if any, of said Filter Pattern,
23 if any, associated to said Class Population Presentation Pattern, said
24 identification by the alias of each of said Filter Variables, and to present
25 said Filter Variables to said user in the order they are defined in said
26 Filter Pattern;
27 o a mechanism to present said user a default value, if any, of each Filter
28 Variable, if any, of said Filter Pattern, if any, associated to said Class
29 Population Presentation Pattern;
30 o a mechanism to let said user provide a value for each Filter Variable, if
31 any, of said Filter Pattern, if any, associated to said Class Population
32 Presentation Pattern;

33 o a mechanism to validate the value assigned to each Filter Variable, if any,
34 of said Filter Pattern, if any, associated to said Class Population
35 Presentation Pattern, said validation carried out according to:
36 • the data type of said Filter Variable;
37 • what is dictated by the Introduction Pattern, if any, associated to
38 said Filter Variable;

39 o a mechanism to access the objects selection mechanism corresponding
40 to a Population Selection Pattern which defines the process of observing
41 and selecting objects in a multiple objects society, said Population
42 Selection Pattern being associated to every object valuated Filter
43 Variable, if any, of said Filter Pattern, if any, associated to said Class
44 Population Presentation Pattern;

45 o a mechanism to present said user with a help message, if any,
46 associated to each Filter Variable, if any, of said Filter Pattern, if any,
47 associated to said Class Population Presentation Pattern;

48 o a mechanism to present said user with elements in any Display Set
49 Pattern assigned as a Supplementary Information Pattern to every object
50 valuated Filter Variable, if any, of said Filter Pattern, if any, associated to
51 said Class Population Presentation Pattern, whenever the value of every
52 said Filter Variable changes;

53 o a mechanism to invoke execution of the query represented by said Filter
54 Pattern, if any, associated with said Class Population Presentation
55 Pattern, or to invoke the retrieval of the full population of said class
56 owning said Class Population Presentation Pattern;

57 - translating said validated formal language specification so as to provide
58 code which controls a computer to provide a mechanism to display the
59 value of each element, for every instance in the population of the class or
60 for every instance returned as a result of executing said query
61 represented by said Filter Pattern, if any, associated with said Class
62 Population Presentation Pattern, said each element being in the Display Set
63 Pattern associated to said Class Population Presentation Pattern, said each
64 element the value of which is displayed being restricted to those elements
65 corresponding to attributes the logged on user is allowed to query;

66 - translating said validated formal language specification so as to provide
67 code which controls a computer to provide a mechanism to select one of
68 the objects or instances presented to the user by said Class Population
69 Presentation Pattern;
70 - translating said validated formal language specification so as to provide
71 code which controls a computer to provide a mechanism to, upon selection
72 of one of said objects presented by said Class Population Presentation
73 Pattern, access each of said user interface interaction scenarios
74 corresponding to each service that can be executed on said selected
75 object, each of said services identified by the alias of the Service
76 Presentation Pattern corresponding to each of said services, said services
77 further restricted to the services the logged on user is allowed to execute;
78 - translating said validated formal language specification so as to provide
79 code which controls a computer to provide a mechanism to present said
80 user with any help message associated with each service executable on
81 said selected object;
82 - translating said validated formal language specification so as to provide
83 code which controls a computer to provide a mechanism to, upon selection
84 of one of said objects presented by said Class Population Presentation
85 Pattern, access each of said user interface interaction scenarios
86 corresponding to each class, if any, owning instances related with said
87 selected object of said class owning said Class Population Presentation
88 Pattern, each of said user interface interaction scenarios being identified by
89 the alias of its corresponding Presentation Pattern, said user interface
90 interaction scenarios being further restricted to the ones corresponding to
91 classes said logged on user is allowed to query;
92 - translating said validated formal language specification so as to provide
93 code which controls a computer to provide a mechanism to present said
94 user any help message associated with each of said user interface
95 interaction scenarios corresponding to each class, if any, owning
96 instances related with said selected object of said class owning said Class
97 Population Presentation Pattern; and
98 - translating said validated formal language specification so as to provide

99 code which controls a computer to provide a mechanism to cancel the
100 interaction of the logged on user with the interface interaction scenario
101 represented by said Class Population Presentation Pattern.

1 63. [NEW] The process of claim 59 wherein the steps in step D of translating said
2 validated formal language specification to generate computer readable code which can
3 control a computer to allow a logged on user to interact with an interaction scenario
4 corresponding to a Master/Detail Presentation pattern so as to query information on:
5 - instances belonging to the class owning said Master/Detail Presentation
6 Pattern as presented in a Presentation Pattern referred to as a Master
7 Presentation Pattern, which is either an Instance Presentation Pattern or a
8 Class Population Presentation Pattern owned by the same class owning
9 said Master/Detail Presentation Pattern, and
10 - instances belonging to other classes related to said class owning said
11 Master/Detail Presentation Pattern, each set of related instances as
12 presented by a Presentation Pattern referred to as Detail Presentation
13 Pattern, which is either an Instance Presentation Pattern, or a Class
14 Population Presentation Pattern, or belonging to a Master/Detail Presentation
15 Pattern owned by said class related with said class owning said Master
16 Presentation Pattern,
17 said steps of translating comprising the steps of:
18 - translating said validated formal language specification so as to provide
19 code which controls a computer to provide a mechanism to identify said
20 Master/Detail Population Presentation Pattern by its alias;
21 - translating said validated formal language specification so as to provide
22 code which controls a computer to provide a mechanism to present said
23 user any help message associated with said Master/Detail Presentation
24 Pattern;
25 - translating said validated formal language specification so as to provide
26 code which controls a computer to provide a mechanism to present said
27 user the Master Presentation Pattern of said Master/Detail Presentation
28 Pattern;
29 - translating said validated formal language specification so as to provide

30 code which controls a computer to provide a mechanism to present said
31 user with every Detail Presentation Pattern of said Master/Detail
32 Presentation Pattern;
33 - translating said validated formal language specification so as to provide
34 code which controls a computer to provide a mechanism to synchronize
35 information displayed in every Detail Presentation Pattern whenever:
36 o the object displayed in the Master Presentation Pattern changes, or
37 o the selection of one of the objects displayed in the Master Presentation
38 Pattern changes
39 - translating said validated formal language specification so as to provide
40 code which controls a computer to provide a mechanism to cancel the
41 interaction of the logged on user with the interaction scenario represented
42 by said Master/Detail Presentation Pattern.
43
44

30 code which controls a computer to provide a mechanism to present said
31 user with every Detail Presentation Pattern of said Master/Detail
32 Presentation Pattern;
33 - translating said validated formal language specification so as to provide
34 code which controls a computer to provide a mechanism to synchronize
35 information displayed in every Detail Presentation Pattern whenever:
36 o the object displayed in the Master Presentation Pattern changes, or
37 o the selection of one of the objects displayed in the Master Presentation
38 Pattern changes
39 - translating said validated formal language specification so as to provide
40 code which controls a computer to provide a mechanism to cancel the
41 interaction of the logged on user with the interaction scenario represented
42 by said Master/Detail Presentation Pattern.

43

44

45

46

47 Dated: February 10, 2006

48

49

50

51

52

53

Respectfully submitted,



Ronald Craig Fish

Reg. No. 28,843

Tel 408 778 3624

FAX 408 776 0426

I hereby certify that this correspondence is being deposited with the United States Postal Service as First Class Mail, postage prepaid, in an envelope addressed to: Mail Stop Amendment, Commissioner for Patents , P.O. Box 1450, Alexandria, Va. 22313-1450.

on 2/10/2006

(Date of Deposit)



Ronald Craig Fish, President

Ronald Craig Fish, a Law Corporation

Reg. No. 28,843